Low Latency Low Loss Scalable Throughput (L4S)

Bob Briscoe, CableLabs
Koen De Schepper, NOKIA Bell Labs
Olga Bondarenko, simula
Ing-jyh (Inton) Tsang, NOKIA

TSVWG, IETF-102, Jul 2018
L4S Recap

- **Motivation**
  - Extremely low queuing delay for all Internet traffic, including link saturating
  - already 1-2 orders better than state of the art
  - 500 μs vs 5-15 ms (fq-CoDel or PIE)

- **Architecture**

![Diagram showing L4S and Classic sender behavior](image)

- **Host**:
  - Scalable sender
  - $r \propto 1/p$

- **Protocol [ECN]**:
  - Classic Classifier
  - Classic: [X0]
  - L4S: [X1]
  - $r \propto 1/\sqrt{p}$

- **Network: DualQ Coupled AQM**:
  - L4S marker
  - Coupling $p^2$
  - Classic drop or marking
  - Strict priority scheduler

$r$: packet rate
$p$: drop/mark probability
L4S draft updates this IETF cycle

tsvwg

- Three core L4S WG drafts in tsvwg
  - L4S Internet Service: Architecture
draft-ietf-tsimg-l4s-arch-03 (-02) [stable]
  - Identifying Modified ECN Semantics for
    Ultra-Low Queuing Delay (L4S)
draft-ietf-tsimg-ecn-l4s-id-05 (-03)
  - DualQ Coupled AQMs for L4S
draft-ietf-tsimg-aqm-dualq-coupled-08 (-06)

- L4S-related individual drafts in tsvwg
  - Identifying and Handling Non-Queue-Building
    Flows in a bottleneck link
draft-white-tsimg-nqb-00 [new]
  - Interactions between L4S and Diffserv
draft-briscoe-tsimg-l4s-diffserv-02 (-01)

Outside tsvwg

- tcpm, implementation, etc

Various Heads-ups

Later talk

Later talk

Complete restructure
Made consistent with other 2 drafts
Comprehensive rework of 'Other IDs'
TCP-RACK-like requirement (previous cycle)

Extra normative requirements
Fixed rigour of maths
Management requirement details
Generalized L4S AQM: step to ramp
Shared vs. dedicated buffers

Do not share this material with anyone other than CableLabs Members, and vendors under CableLabs NDA if applicable.
Identifying Modified ECN Semantics for Ultra-Low Queuing Delay (L4S)

draft-ietf-tsvwg-ecn-l4s-id-05
Complete restructure (-03 to -04)

BEFORE:

2. L4S Packet Identifier ...........................................

2.1. Consensus Choice of L4S Packet Identifier: Requirements

2.2. L4S Packet Identification at Run-Time .................

2.3. Interaction of the L4S Identifier with other Identifiers

2.4. Pre-Requisite Transport Layer Behaviour ..............

2.4.1. Pre-Requisite Congestion Response .................

2.4.2. Pre-Requisite Transport Feedback ...............  

2.5. Exception for L4S Packet Identification by Network Nodes

     with Transport-Layer Awareness ....................

2.6. The Meaning of L4S CE Relative to Drop ............

AFTER:

2. Consensus Choice of L4S Packet Identifier: Requirements ...........................................

3. L4S Packet Identification at Run-Time ......................

4. Prerequisite Transport Layer Behaviour ..................

4.1. Prerequisite Codepoint Setting ......................

4.2. Prerequisite Transport Feedback ....................

4.3. Prerequisite Congestion Response .............

5. Prerequisite Network Node Behaviour .................

5.1. Prerequisite Classification and Re-Marking Behaviour

5.2. The Meaning of L4S CE Relative to Drop ........

5.3. Exception for L4S Packet Identification by Network Nodes

     with Transport-Layer Awareness ..................

5.4. Interaction of the L4S Identifier with other Identifiers

• Only structure changes
  • Text unchanged (except to introduce structure)

• Collected together:
  • Transport Requirements
  • Network Requirements
Other Identifiers (-04 to -05)

- Default classifier on 2-bit ECN field in IP header (v4 or v6)
  - if ECT(1) or CE, forward to L4S

<table>
<thead>
<tr>
<th>Codepoint</th>
<th>ECN bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not-ECT</td>
<td>00</td>
</tr>
<tr>
<td>ECT(0)</td>
<td>10</td>
</tr>
<tr>
<td>ECT(1)</td>
<td>01</td>
</tr>
<tr>
<td>CE</td>
<td>11</td>
</tr>
</tbody>
</table>

- Eg.1) Inclusion
  - AND optionally
  - Later talk (Non-Queue-Building)

- Eg.2) Exclusion
  - BEFORE optionally

- Add traffic into L queue
- MUST be compatible with L4S
- Classifier on any other field
  - source or dest. IP address, VLAN ID
  - L7 protocol (e.g. DNS, LDAP)
  - Local or Global DSCP (e.g. EF, VA, NQB)

- Exclude traffic from L queue
- Depends on local policy
  - security: e.g. malicious hosts
  - commercial: e.g. lower-tier customers
- Local-use classifiers only
  - addresses, local-use DSCPs
Other Identifiers: within a Diffserv queuing hierarchy

- Previous examples split Default class (BE) into two
- Operator may want to offer additional bandwidth priority services
  - not usually necessary for public Internet
  - beyond scope of core L4S drafts

- For ecn-l4s-id, the important points are:
  - Global or Local-use DSCPs
  - Two main classification types:
    - PHBs before DualQ (eg.3)
    - PHBs after one of the DualQs
    - or both

Later talk (l4s-diffserv)
5th Requirement for L4S senders

- 'TCP Prague' Requirements (for all transports, not just TCP)
  draft-ietf-tsvwg-ecn-l4s-id-05#section-4.3

- to use ECT(1), a scalable congestion control MUST detect loss:
  - by counting in units of time
  - not in units of packets

- Then link technologies that support L4S can remove head-of-line blocking delay
  - see talk in tsvwg-IETF-102 or tcpm later today (or Appendix A.1.7)

- This has raised a more general deployment question...
Could L4S Get Stuck on DCTCP?

- MUST comply with TCP Prague requirements for public Internet
  - for everyone to gain benefits
- But claimed that L4S can be tested / trialled with DCTCP (non-compliant on #2-#5)
- So how does a network move from trial (with DCTCP) to production (without)?
- Various possible answers
  - gradually?
  - deploy queue protection / policing?
  - depends on the requirement
- Specifically
  1) Fall back to Reno-friendly on drop
  2) Fall back to Reno-friendly on Classic ECN AQM
  3) Remove RTT bias
  4) Scale cwnd below 2 SMSS
  5) Detect loss in units of time

Flow throughput ‘fairness’ issues (deploy policers?)
Queuing delay issue (deploy queue protection?)
If links turn off resequencing to scale, more DCTCP spurious re-xmts
DualQ Coupled AQMs for L4S

draft-ietf-tsvwg-aqm-dualq-coupled-08
Extra normative requirements

- Previous normative requirements were necessary but not sufficient
- A Dual Queue Coupled AQM implementation MUST utilize two queues, each with an AQM algorithm.*
- The AQM algorithm for the low latency (L) queue MUST apply ECN marking.
- A DualQ Coupled AQM MUST apply ECN marking to traffic in the L queue that is no lower than that derived from the likelihood of drop (or ECN marking) in the Classic queue using Eqn. (1).**
- a parameter for typical or target queuing delay in each queue [...] MUST be expressed in units of time.

* Can be part of a larger queuing hierarchy
** Equations have been re-worked:
    Previously instantaneous marking was equated to stationary marking
Management Requirements: Added Details

• Queue delay measurement
  To facilitate comparative evaluation of different implementations and approaches, an implementation SHOULD allow mean and 99th percentile queue delay to be derived
    • Suggested coarse histogram method with configurable bin edges

• Overload reporting
  • Suggested a hysteresis method to prevent flapping in and out of overload causing event storms

• Checked against RFC5706 (Ops & Mgmt req's for experiments)
DualPI2 Pseudocode Appendix

- L4S AQM: generalized from step to ramp
  - Initial experiments no worse than step
  - Will enable experiments with faster convergence of 'TCP Prague'

- Dedicated buffers vs. shared: Pros and Cons
  - better isolation from tail drop due to large C bursts
  - less memory efficient (given L rarely uses much)
L4S status update (1/2)

- Landing page for code, specs, papers
  
  https://riteproject.eu/dctth/

- Source Code
  - Dual Queue Coupled AQM, DualPI2 for Linux [UPDATE] – new API (parameter independence), overload protection, in non-overload conditions no performance impact
  - Data Centre TCP (DCTCP) for Linux – traced rounding bug in EWMA – fix to be posted
  - Accurate ECN TCP Feedback for Linux [testing needed]

- Implementations
  - DualQ Coupled AQM: in at least one chipset aimed at DC environment [availability TBA]
  - L4S Scalable congestion control: rmcat SCReAM
  - BBRevo, evolution of BBR with L4S support
  - Whole L4S system in ns3 [complete but evolving]
L4S status update: IETF specs (2/2)
Deltas since last IETF in Red

tsvwg
- L4S Internet Service: Architecture <draft-ietf-tsvwg-l4s-arch-03> [stable]
- Identifying Modified ECN Semantics for Ultra-Low Queuing Delay (L4S) <draft-ietf-tsvwg-ecn-l4s-id-05> [2 UPDATES]
- DualQ Coupled AQMs for L4S: <draft-ietf-tsvwg-aqm-dualq-coupled-08> [2 UPDATES]
- Interactions of L4S with Diffserv <draft-briscoe-tsvwg-l4s-diffserv-02> [UPDATE]
- Identifying and Handling Non-Queue-Building Flows in a bottleneck link draft-white-tsvwg-nqb-00 [NEW]
- enabled by <RFC8311> [RFC published]

tcpm
- scalable TCP algorithms, e.g. Data Centre TCP (DCTCP) <RFC8257>, TCP Prague
- Accurate ECN: <draft-ietf-tcpm-accurate-ecn-07>
- ECN++ Adding ECN to TCP control packets: <draft-ietf-tcpm-generalized-ecn-03> [UPDATE]

Other
- ECN support in trill <draft-ietf-trill-ecn-support-07>, motivated by L4S [RFC Ed Q]
- ECN in QUIC <draft-ietf-quic-transport-16>, [motivated by L4S – 3 Updates, but not ECN part]
- ECN and Congestion Feedback Using the Network Service Header (NSH) <draft-eastlake-sfc-nsh-ecn-support-02> [UPDATE] [supports L4S-ECN]
Next Steps for 3 core L4S drafts

- Can now leave holding pattern
  - sufficient progress on TCP Prague requirements within the stable architecture
- Tidied up 3 years of piecemeal changes
- Invited reviews in progress – need more
- Ready for WGLC
  - target Dec'18 – or Jan'19